aval Research Laboratory
shington, DC 20375-5000

NRL Memorandum Report 6740

Noise Effects Upon A Simple Timing Channel

IRA S. MOSKOWITZ

Center for Secure Information Technology
Information Technology Division

December 21, 1990

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	1990 December 21	Interim

4. TITLE AND SUBTITLE

Noise Effects Upon A Simple Timing Channel

5. FUNDING NUMBERS

55-2830-0-0
RR015-09-41 (WU)
6.1

6. AUTHOR(S)

Ira S. Moskowitz

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Research Laboratory
Washington, DC 20375-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

NRL Memorandum
Report 6740

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
Arlington, VA 22217

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

We investigate the effects of noise upon a covert timing channel in a multi-user computer system. Shannon's information theory is used to quantify the resulting information flow across the channel. Possible Trojan Horse strategies are also discussed.

14. SUBJECT TERMS

Computer security
Information theory
Covert channels

15. NUMBER OF PAGES

22

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR

CONTENTS

NOISE EFFECTS UPON A SIMPLE TIMING CHANNEL

1 Introduction

We take as our area of interest a multi-user computer system. Of special interest are two users called *high* and *low*. We assume that security procedures have been set up so that low may not read high's files and high may not write its files to low. These are the no read-up, no write-down requirements of the Bell-La Padula model [1] of security. However, it is possible for high to interfere with the system response time to low's input. We will only be interested in delays to low's input of a specific query designated by ?. The interference that high is causing might not be a conscious act of the high user. It may be attributable to a surreptitious act performed by a program referred to as a *Trojan Horse*. The Trojan Horse has the ability to instigate the delaying procedure to low, read high's file, code it and pass it through this delaying procedure.

Such a means of communication between high and low, with or without the Trojan Horse in place, will be referred to as a **Covert Timing Channel**, or more succinctly, a **Timing Channel**. In [5] Millen discusses a simple timing channel where a reply takes one tick (time unit) if high is not interfering with low, and two ticks if high is interfering. One tick tells the low user to interpret the message as a 0 and two ticks as a 1. Millen restricted his investigations to noiseless channels. We will look at a similar situation as his, except that noise will be introduced into the transmission process.

The noise effects in our model are envisioned as being due to the time sharing of the CPU and I/O delays in our system. The noise effects are not viewed as being attributable to disk read or write delays as in [3].

The noise that we will be looking at will not affect the value of the output as in [6]. The noise will only affect the *timing* of the output. Instead of exactly knowing when the signal will arrive at the low user, the users will have knowledge only of the arrival times in a probabilistic sense. (We are assuming, for the sake of simplicity, that a Trojan Horse having knowledge of arrival times is synonymous with the users having that knowledge also.) We feel that this is a realistic model in light of response time degradation due to many users being on the system simultaneously. We will refer to this as *contention induced noise*. Wittbold and Johnson [9] have also looked at contention for resources as a noise effect but not in this manner. We will use the exponential distribution to model the uncertainty in arrival times of signals to the low user. Let us recall the definition of the exponential distribution.

Manuscript approved July 23, 1990

Definition 1 *The* **exponential random variable**, *with parameter* $\lambda > 0$, *has the probability density function*

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{if } x \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

If our system were noiseless there would be no degradation of the arrival time of a signal. In this case we could say that a signal arrives exactly at time 0. (Without loss of generality we can view 0 as the arrival time when the input was sent at some "negative time". However, if there is contention for resources, which is the case that we are interested in, then we will model the output signal to arrive exponentially distributed in time with parameter λ. This will be the major assumption in this paper.

Now we will slightly modify our probability distribution by a change of variable, to have the arrival time be more physically realistic. Suppose that low does his/her input query ? at time 0. In Millen's noiseless system the output will arrive at time 1. In our noisy system the output will arrive via an exponential distribution starting 1 tick after ?. If high is interfering with low then the output will arrive via an exponential distribution starting 2 ticks after ?.

This leads us to formalize these ideas with the following definition.

Definition 2 *If high is not interfering with low then the response to ?, inputted at time* 0, *is given by the distribution with probability density function*

$$f_1(t) = \begin{cases} \lambda e^{-\lambda(t-1)} & \text{if } t \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

and if high is interfering with low then the response to ? is given by

$$f_2(t) = \begin{cases} \lambda e^{-\lambda(t-2)} & \text{if } t \geq 2 \\ 0 & \text{otherwise.} \end{cases}$$

This is a noisy version of what Millen did in [5]. By letting the parameter λ approach ∞ we can arrive at the same situation that Millen set up. The parameter λ can be adjusted for the amount of noise in the system to demonstrate different possible scenarios with regard to contention induced noise.

The expectation of a random variable with density function

$$f_\tau(t) = \begin{cases} \lambda e^{-\lambda(t-\tau)} & \text{if } t \geq \tau \\ 0 & \text{otherwise} \end{cases}$$

is given by

$$E_\tau = \int_{-\infty}^{\infty} t f_\tau(t)\, dt = \int_{\tau}^{\infty} \lambda t e^{-\lambda(t-\tau)}\, dt\ .$$

The expectation can be interpreted as the average time that the output signal will arrive at the low users terminal. By integration by parts we see that

$$E_\tau = \tau + \frac{1}{\lambda}$$

which is not surprising. This shows that as λ approaches ∞ that the average waiting time for a response to ? is τ. We can also see that the expectation, up to an additive constant, is directly proportional to the noise and is inversely proportional to λ. This relationship is important as we will see later when we discuss mutual information and channel capacity.

2 Strategy

Both the high and low user have knowledge of the stochastic processes that are involved in the timing channel. The users have to decide upon a strategy [8] to exploit the communication channel that now exists.

The high user wishes to send a binary file to the low user across the covert communication channel. (Again this may be intentional on high's part or due to the action of a Trojan Horse.) If a 0 is to be sent, then high will not interfere with low and the arrival time of the output from ? is given by the probability density function f_1. If high wishes to send a 1, then it will interfere with low and the arrival time of the output in response to ? will be given by the probability density function f_2. Of course the way things stand now, high must have some feedback to know whether or not low received the output. So it is necessary to have some sort of high level auditing going on [4, 9]. Due to the probabilistic nature of the response time to ? we have an unbounded possible response time. Of course by adjusting λ we can lower the probability that the output will arrive after some given finite amount of time. But we can never say with certainty that the output will not arrive after the above mentioned finite time. We must make some adjustments in the strategy so that a feasible and realistic communication channel is, in fact, set up between the high and low users.

Let κ represent the time that the signal arrives after ? is inputted. Without any restrictions we have that $1 \leq \kappa < \infty$. As discussed above this can lead us into a situation where low has an infinite wait for output to ?.

Strategy: *Low will input ? every 2 ticks. High will either interfere or not. If* $1 \leq \kappa < 2$ *then low will interpret* κ *to be a* 0. *If* 2 *ticks have gone by on low's clock, then low will automatically assume that the message is a* 1 *and issue an interrupt to its previous query ? before inputting its next query ?.*

A pleasing aspect of this strategy is that high does not have to audit low's signal. However, high does know the strategy that is being used and can code its message accordingly. So high, or the Trojan Horse, codes the file that it wants to send and every 2 ticks it will either interfere with low or not. Low, or the other part of the Trojan Horse, will interpret the signal that it gets according to the strategy. We see that it takes 2 ticks for a symbol to be transmitted over the channel. We refer to this as a *cycle*. Even though high need not be able to audit low it is necessary for both high and low to know when low will be inputting its query. It is possible that the Trojan Horse is designed to work at certain times or that high has a limited audit ability to know when low will start inputting ? and how many cycles low will keep doing this for.

The reason that low must issue an interrupt, if it has not yet received a response to ?, is to prevent a response from "leaking" over into the next cycle of query and reply. Say for example that low inputs ?, 2 ticks go by and no response is given by the system, and then low again inputs ?. How is low to know when it finally does receive a response if it is the response to the first ? or the second ?? The issuance of an interrupt after 2 ticks will preclude this situation. We assume that the interrupt stops the response to ? from reaching the low user.

The same strategy without the interrupts is an interesting and complicated problem. In fact, the channel is still memoryless. We hope to address this situation in future work.

3 Transmission Errors

There are obvious transmission errors in our strategy which result in signal noise. There are no errors if high sends a 1 because $2 \leq \kappa$. The low user is watching his/her clock and as soon as 2 ticks have gone by low interprets the message as a 1. However, if high wishes to send a 0, then errors can be introduced. If κ arrives before 2 ticks have elapsed there is no transmission error; however, if $2 \leq \kappa$ then we do have an error because low will interpret the signal as a 1 when it in fact is a 0. The probability

of a 0 being sent and a 0 being received is

$$P(0_r \mid 0_s) = \int_1^2 \lambda e^{-\lambda(t-1)}\, dt = 1 - e^{-\lambda} \quad . \tag{1}$$

The probability of a 0 being sent and a 1 being received is

$$P(1_r \mid 0_s) = \int_2^\infty \lambda e^{-\lambda(t-1)}\, dt = e^{-\lambda} \quad . \tag{2}$$

As discussed above we also have that

$$P(1_r \mid 1_s) = 1 \text{ and } P(0_r \mid 1_s) = 0. \tag{3}$$

We will use equations (1), (2), and (3) to calculate the channel capacity of the covert timing channel.

4 Information Theory

4.1 Shannon's Work

We will use Shannon's [7] information theory to investigate how much "information" can be sent over the covert channel in question. Let us review some of the concepts of information theory. The channels in this paper are *memoryless*, meaning that each occurrence of inputting a symbol is independent from the previous occurrences.

We have a communication channel and we let Ω stand for the input random variable and Ψ represent the output random variable. In our example X, the signal that high is trying to pass to low will be the input variable, and Y the signal that low actually receives, will be the output variable. The alphabet of both X and Y is the set $\{0,1\}$. We are assuming that high is trying to send a binary file that has an equal distribution of 0's and 1's. This leads us to the fact that $P(X = 0) = P(X = 1) = 1/2$. We will use the shorthand notation $P(X = i) = P(x_i) = P(i_s)$ and $P(Y = i) = P(y_i) = P(i_r)$.

If Ω and Ψ are discrete random variables taking values in the alphabets $\{\omega_i\}$ and $\{\psi_j\}$, respectively, we may define the entropy of Ω to be $E(I(\Omega))$. (Note that all logs are base 2, and I is self-information.)

Definition 3 *The* **entropy** *of* Ω *is* $H(\Omega) = -\sum_i P(\omega_i) \log P(\omega_i)$.

The entropy measures the amount of information that Ω is conveying on the average. When we view Ω as a random variable in a communication channel we measure the entropy in bits per (transmission) symbol. So we see that $H(X) = -P(0_s) \log P(0_s) - P(1_s) \log P(1_s) = -(1/2 \log 1/2 + 1/2 \log 1/2) = 1$. It is not a surprising result that X "sends" 1 bit of information, on the average, each time it transmits either a 0 or a 1. In fact for a Bernoulli random variable it is easily shown that the maximum of the entropy is 1.

We wish to find out how much information is transmitted (on the average) across the communication channel. In other words, how much information is the low user receiving from the high user? One cannot use $H(\Psi)$ alone to answer this if we have a noisy channel. We have to see how the output to Ψ depends on what Ω is inputting and how the noise induced by the probabilistic uncertainty influences the receiver's (low's) output. In Shannon's words [7], the equivocation "measures the average ambiguity of the received signal". Formally:

Definition 4 *Define the* **equivocation** *to be*

$$H_\Psi(\Omega) = -\sum_j \sum_i P(\psi_j) P(\omega_i \mid \psi_j) \log P(\omega_i \mid \psi_j).$$

We still have to define a term that measures the average information sent over the channel — this is given by the mutual information:

Definition 5 *Define the* **mutual information** *between* Ω *and* Ψ *to be*

$$I(\Omega, \Psi) = I = H(\Omega) - H_\Psi(\Omega). \tag{4}$$

Both the equivocation and the mutual information are measured in bits per symbol. By symbol we mean the process of Ω inputting a signal and Ψ in turn receiving a signal.

We wish to find out how much information is being sent over the communication channel. This gives a measurement of how the channel behaves over a long period of time, this long period of time being when the process — of low inputting ?, high interfering or not, and low receiving output — is repeated many times. The measurement that we are looking for is the channel capacity. The channel capacity in its most

general form involves looking at the limiting behavior of the mutual information as we try to maximize over different probability distributions associated to input random variables with identical alphabets. For a memoryless channel we do not have to worry about the limiting behavior as long as we measure this channel capacity in terms of bits/symbol because the past history of the channel does not influence the capacity calculation.

Definition 6 *The* **channel capacity**, *or simply the* **capacity**, *of a memoryless communication channel is given by*

$$C = \sup_{\mu} I(\Omega, \Psi)$$

Where μ are the different probability measures associated to all random variables that have the same alphabet as Ω.

To be precise we are actually taking the supremum over the different random variables with alphabets identical to that of Ω's. So the notation $I(\Omega, \Psi)$ is actually incorrect in the definition of capacity. But it is a common abuse and we will be consistent with (some) history. In other words, for the purposes of the definition of capacity we are actually viewing Ω as a family of random variables.

The interested reader can see the "noisy example" in [6] for a simple example involving a binary symmetric channel with a cross-over probability.

4.2 Analysis

We will compare this to a noiseless situation where if high wishes to send a 0, low receives the signal after one tick and if high wishes to send a 1, low receives the signal after two ticks; thus whatever symbol that is passed the cycle time is 2 ticks. The channel capacity in this example is 1 bit per 2 ticks or just simply .5 bits/tick.

Now we are considering the noisy case where $0 < \lambda$. As discussed earlier $H(X) = 1$. The equivocation is

$$\begin{aligned} H_Y(X) \;=\; & -\{P(0_r)P(0_s \mid 0_r)\log P(0_s \mid 0_r) + P(0_r)P(1_s \mid 0_r)\log P(1_s \mid 0_r) \\ & + P(1_r)P(0_s \mid 1_r)\log P(0_s \mid 1_r) + P(1_r)P(1_s \mid 1_r)\log P(1_s \mid 1_r)\}. \end{aligned}$$

By using our assumption that $P(0_s) = P(1_s) = .5$ and Bayes' formula we may calculate all of the probabilities required for the equivocation. Let us summarize all of the probability values below.

$$\begin{aligned}
P(0_s) &= .5 \\
P(1_s) &= .5 \\
P(0_r \mid 0_s) &= 1 - \epsilon^{-\lambda} \\
P(1_r \mid 0_s) &= \epsilon^{-\lambda} \\
P(1_r \mid 1_s) &= 1 \\
P(0_r \mid 1_s) &= 0 \\
P(0_r) &= (1 - \epsilon^{-\lambda})/2 \\
P(1_r) &= (1 + \epsilon^{-\lambda})/2 \\
P(0_s \mid 0_r) &= 1 \\
P(1_s \mid 0_r) &= 0 \\
P(0_s \mid 1_r) &= \epsilon^{-\lambda}/(1 + \epsilon^{-\lambda}) \\
P(1_s \mid 1_r) &= 1/(1 + \epsilon^{-\lambda})
\end{aligned}$$

Therefore, the equivocation is

$$H_Y(X) = -\frac{1}{2}[\epsilon^{-\lambda} \log \epsilon^{-\lambda} - (1 + \epsilon^{-\lambda}) \log(1 + \epsilon^{-\lambda})].$$

It is easy to see that :

$$\lim_{\lambda \to 0^+} H_Y(X) = 1 \text{ and } \lim_{\lambda \to \infty} H_Y(X) = 0.$$

Consider these facts in light of (4). These facts corresponds to the fact that as λ increases the noise is decreasing. The situation where $\lambda = \infty$ is the case of no contention induced noise and we can view the probability distributions for $f_1(t)$ and $f_2(t)$ as the Dirac functions $\delta(t-1)$ and $\delta(t-2)$, respectively. This is the exact situation that Millen described. (Remember that his channel capacities are in terms of bits/tick, ours are, at this point, being expressed in terms of bits/symbol.) As $\lambda \to 0$ the noise is increasing and the arrival time for the symbol 0 at the receiver is moved more and more away from 1 tick towards 2 ticks. This is due to the fact that

as λ diminishes the maximum of $f_1(t)$ diminishes but its rate of asymptotic approach to 0 as $x \rightarrow \infty$ slows.

The mutual information is

$$I(X,Y) = 1 + \frac{1}{2}[\epsilon^{-\lambda} \log \epsilon^{-\lambda} - (1+\epsilon^{-\lambda}) \log(1+\epsilon^{-\lambda})]. \tag{5}$$

Figure 1 is a plot of $I(X,Y)$

We see that $I(X,Y)$ is asymptotic to 1 as $\lambda \rightarrow \infty$. We must remember that the mutual information I is less than the channel capacity. However, the information is passed at a rate equal to I if no coding is done. Shannon's theorem tells us that we can achieve a transmission rate within ϵ of I by the proper coding [7]. This does not tell us what the code is. So if our Trojan horse does not know the code then it is restricted to a transmission rate equal to I.

The mutual information acts as a lower bound for the channel capacity C. A natural question arises as to how much "faster" C is than I. If the difference is negligible than we might as well just restrict our attention to I and not get into coding issues. However, if $I << C$, then we should do some analysis involving C because then I would give the person interested in security too optimistic a view as to the leakage capabilities of the covert timing channel that exists in the system.

Let us calculate the actual channel capacity. To do this we will recalculate $I(X,Y)$ for a general value of $P(0_s)$ instead of just $1/2$. We will then view I as a function of p. The mutual information function $I(p)$ is concave down [2, Thm. 5.2.5] with respect to the variable p. Hence, it suffices to find a critical point. In other words we will solve

$$I'(p) = 0. \tag{6}$$

Let us list the various probabilities that will be needed to solve for I:

$$\begin{aligned}
P(0_s) &= p \\
P(1_s) &= 1-p \\
P(0_r \mid 0_s) &= 1-e^{-\lambda} \\
P(1_r \mid 0_s) &= e^{-\lambda} \\
P(1_r \mid 1_s) &= 1 \\
P(0_r \mid 1_s) &= 0 \\
P(0_r) &= (1-e^{-\lambda})p \\
P(1_r) &= (e^{-\lambda}-1)p+1
\end{aligned}$$

$$
\begin{aligned}
P(0_s \mid 0_r) &= 1 \\
P(1_s \mid 0_r) &= 0 \\
P(0_s \mid 1_r) &= e^{-\lambda}p/(1-p+e^{-\lambda}p) \\
P(1_s \mid 1_r) &= (1-p)/(1-p+e^{-\lambda}p)
\end{aligned}
$$

The input entropy $H(X)$ is $-p\log p-(1-p)\log(1-p)$.

The mutual information $I(p)$ is now:

$$-p\log p+e^{-\lambda}p\log e^{-\lambda}p-(1-p+e^{-\lambda}p)\log(1-p+e^{-\lambda}p). \tag{7}$$

Thus we see that

$$I'(p)=-\log p+e^{-\lambda}\log e^{-\lambda}p+(1-e^{-\lambda})\log(1-p+e^{-\lambda}p). \tag{8}$$

Therefore the zero of $I'(p)$ is when:

$$p=\frac{1}{1+e^{\lambda/(e^{\lambda}-1)}-e^{-\lambda}}$$

which we will set equal to ζ. It is easy to show that:

$$\lim_{\lambda\to\infty}\zeta=1/2.$$

In fact from **Figure 2** we see that the zero of $I'(p)$ quickly becomes asymptotic to $1/2$. This tells us that for relatively moderate choices of λ that $I(X,Y)$ is maximized for an input probability distribution where both 0's and 1's are sent with equal probabilities of $1/2$. So using $I(X,Y)$ as we calculated it in (5) gives a very good approximation to the actual channel capacity.

The actual channel capacity is:

$$-\zeta\log\zeta+e^{-\lambda}\zeta\log e^{-\lambda}\zeta-(1-\zeta+e^{-\lambda}\zeta)\log(1-\zeta+e^{-\lambda}\zeta). \tag{9}$$

Figure 3 is a graph of the channel capacity (9). We can see how similar it is to figure 1. In fact, **Figure 4** is a comparison of the two functions for small λ. Also **Figure 5** is the graph of the difference $C-I(1/2)$. From these figures we see how good an approximation I is to C.

4.3 Security Considerations

How does the above analysis help us in trying to make our computer system secure? If it is possible to quantify the parameter λ then we may be able to get a handle on how bad our covert timing channel is. If we know that during peak usage time of the computer system that λ is very small then we do not have as large a channel as perhaps at night when there are minimal users on the system and λ is high. At least the above gives us a way of measuring relative insecurities due to the timing channel.

Another approach may be to monitor system usage and when λ rises above a certain level, automatic processes may be started to lower λ. Of course this degrades overall system performance but in many cases computer security may not come without the additional burden of diminished performance. By adjusting the background contention to modify λ we have a parameter that we can fine tune. If we are specifying a system that has channels below a certain capacity we can use the above ideas to make sure that our system matches these claims.

5 Conclusion

We have given a model of a noisy timing channel that shows how high can interfere with low through the timing of the output to low's input of a certain query ?. When the noise is too high the communication channel is effectively shut down. When the noise is minimal it behaves as the noiseless situation. Furthermore, we have shown that it is not necessary for high to use the most effective coding of its message. A simple equiprobable coding will suffice to effectively pass the message at a rate very close to the actual channel capacity. This gives us a strategy that can be used in all situations without high having to constantly adjust its coding algorithm in response to new values of λ.

It is hoped in future work to give a model where a cycle is no longer 2 ticks but is the length of time that low actually has to wait for a response, provided it is less than 2 ticks. This of course says that we have to allow a high level audit. We would then like to compare these results to Millen's [5] concerning channel capacity and the golden mean. Also, as discussed earlier, we would like to remove the necessity of low issuing an interrupt.

6 Acknowledgements

We wish to thank Jim Gray and John McLean for their comments.

References

[1] D.E. Bell and L.J. La Padula. *Secure Computer System: Unified Exposition and Multics Interpretation, MTR-2997.* MITRE Corp., Bedford, MA, March 1976.

[2] Richard E. Blahut. *Principles and Practice of Information Theory.* Addison-Wesley, Reading, Massachusetts, 1987.

[3] B.D. Gold, R.R. Linde, R.J. Peeler, M.Schaefer, J.F. Scheid, and P.D. Ward. A security retrofit of vm/370. In *AFIPS Conference Proceedings, 1979 National Computer Conference*, volume 48, pages 335–344, Montvale, NJ, 1979.

[4] John McLean. Security models and information flow. In *Proc. 1990 IEEE Symposium on Security and Privacy*, pages 180–187, Oakland, CA, May 1990.

[5] Jonathan K. Millen. Finite-state noiseless covert channels. In *Proc. The Computer Security Foundations Workshop II*, pages 81–86, Franconia, NH, June 1989.

[6] Ira S. Moskowitz. Quotient states and probabilistic channels. In *Proc. The Computer Security Foundations Workshop III*, pages 74–83, Franconia, NH, June 1990.

[7] Claude E. Shannon and Warren Weaver. *The Mathematical Theory of Communication.* University of Illinois Press, Urbana, IL, 1949.

[8] J. Todd Wittbold. Controlled signalling systems and covert channels. In *Proc. The Computer Security Foundations Workshop II*, pages 87–104, Franconia, NH, June 1989.

[9] J. Todd Wittbold and Dale M. Johnson. Information flow in nondeterministic systems. In *Proc. 1990 IEEE Symposium on Computer Security and Privacy*, pages 144–161, Oakland, CA, May 1990.

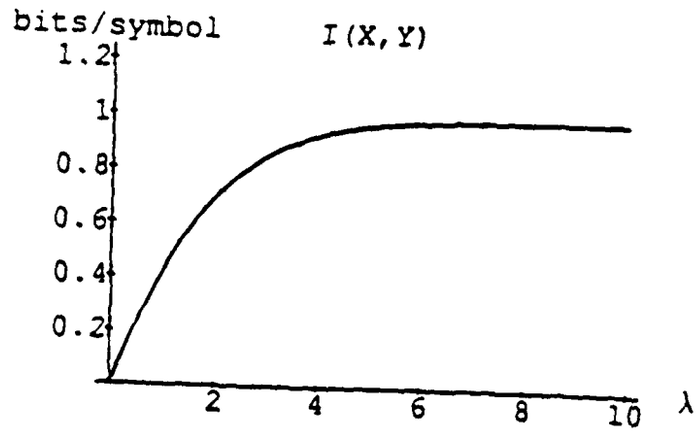

Figure 1

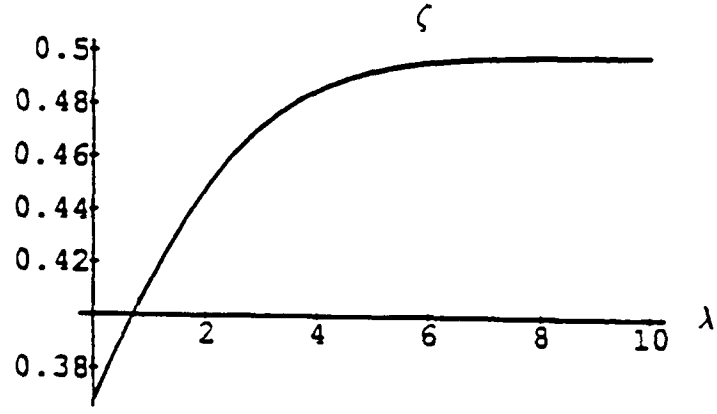

Figure 2

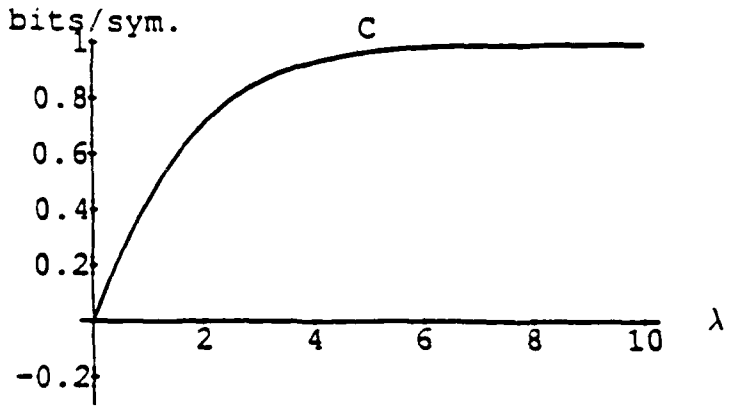

Figure 3

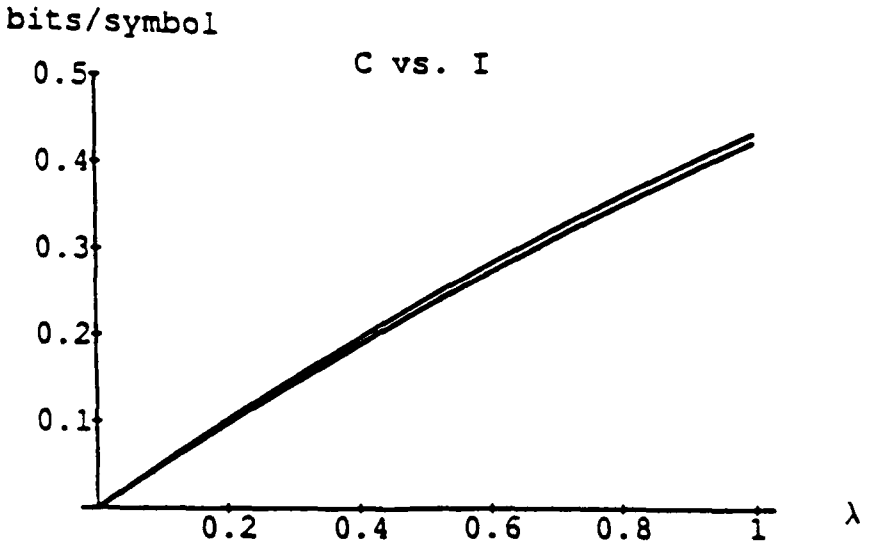

Figure 4

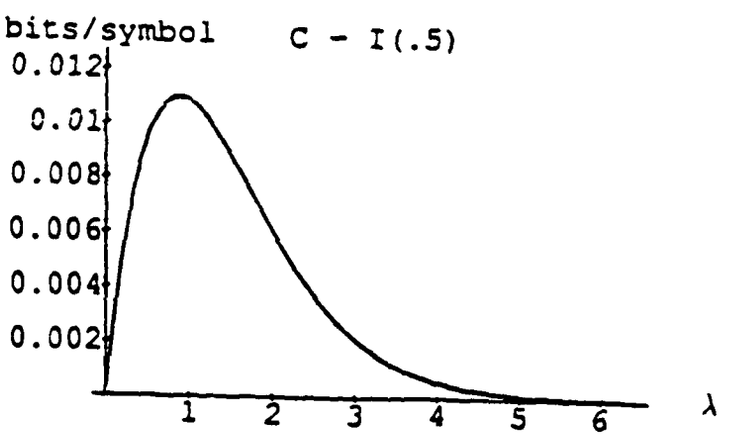

Figure 5